WHAT IS CLAIMED IS:

1. A system for controlling an application process comprising:

an injector;

redirect code operable to be placed in a memory of the application process; and

a library of redirect functions operable to be referenced by the redirect code during the application process execution, the redirect code operable to intercept a set of target function calls made by the application process and execute the redirect functions for the intercepted target function calls.

2. The system, as set forth in claim 1, wherein the injector is pushed to a device executing the application process.

3. The system, as set forth in claim 1, wherein the set of target function calls comprises socket function calls.

4. The system, as set forth in claim 1, wherein the library of redirect functions comprises a dynamic link library.

5. The system, as set forth in claim 1, further comprising:

a secure environment having a plurality of resources;

a firewall securing all access to the plurality of resources in the secure environment; and

an access policy pushed to a device executing the application process, the access policy identifying the resources authorized for access by the device.

6. The system, as set forth in claim 5, wherein the application process comprises an application operable to communicate with the secure environment resources using an Internet transport protocol, the redirect code, and the redirect functions.

7. The system, as set forth in claim 1, wherein the application process comprises an email application.

8. The system, as set forth in claim 1, wherein the application process comprises a web browser application.

9. The system, as set forth in claim 1, wherein the application process comprises a file transfer application.

10. A method for controlling an application process comprising:

pushing an injector to a device executing the application process;

injecting a redirect code into the application process;

executing the redirect code in the application process to reference a redirect library of redirect functions;

resuming the execution of the application process; and

intercepting at least one target function calls made by the application process and executing at least one redirect function in place of the at least one target function calls.

11. The method, as set forth in claim 10, wherein injecting a redirect code further comprises:

starting the application process;

interrupting the execution of the application process; and

injecting the redirect code into a memory space of the application process.

12. The method, as set forth in claim 10, wherein injecting a redirect code further comprises:

starting the application process using a debug option;

catching an exception thrown by the application process;

locating memory space in the application process;

injecting the redirect code into the memory space of the application process; and

set an instruction pointer to the redirect code.

13. The method, as set forth in claim 10, wherein injecting a redirect code further comprises:

starting the application process using a suspend option;

creating memory space in the application process;

injecting the redirect code into the memory space of the application process; and

set an instruction pointer to the redirect code.

14. The method, as set forth in claim 10, wherein injecting a redirect code further comprises:

starting the application process using a suspend option;

creating memory space in the application process;

injecting the redirect code into the memory space of the application process; and

use a create remote thread function to execute the redirect code.

15. The method, as set forth in claim 10, wherein executing the redirect code comprises:

loading the redirect library of redirect functions;

determining a location of an import table replacement function in the redirect library; and

executing the import table replacement function.

16. The method, as set forth in claim 15, wherein loading the redirect library of redirect functions comprises loading a dynamic link library.

17. The method, as set forth in claim 15, wherein executing the import table replacement function comprises:

searching an import table of the application process for the set of target function calls; and

modifying the target function calls to reference redirect functions in the redirect library.

18. The method, as set forth in claim 15, wherein executing the import table replacement function comprises:

searching dynamic link libraries of the application process for the set of target function calls; and

modifying the target function calls to reference redirect functions in the redirect library.

19. The method, as set forth in claim 10, further comprising:

receiving user information;

authenticating the user information;

pushing an access policy specifying resources accessible by a user associated with the user information to a device used by the user.

20. The method, as set forth in claim 19, wherein intercepting at least one target function call comprises intercepting at least one socket function call.

21. The method, as set forth in claim 19, further comprising executing redirect functions to enable a secured access to a plurality of resources via a firewall.

22. A method comprising:

receiving user information;

authenticating the user information;

pushing an injector to a device executing an application process; and

intercepting at least one target function call made by the application process to at least one of a plurality of secure resources and executing at least one redirect function in place of the at least one target function call.

23. The method, as set forth in claim 22, further comprising:

injecting a redirect code into the application process;

executing the redirect code in the application process to reference a redirect library of redirect functions; and

resuming the execution of the application process.

24. The method, as set forth in claim 23, wherein injecting a redirect code further comprises:

starting the application process;

interrupting the execution of the application process; and

injecting the redirect code into a memory space of the application process.

25. The method, as set forth in claim 23, wherein injecting a redirect code further comprises:

starting the application process using a debug option;

catching an exception thrown by the application process;

locating memory space in the application process;

injecting the redirect code into the memory space of the application process; and

set an instruction pointer to the redirect code.

26. The method, as set forth in claim 23, wherein injecting a redirect code further comprises:

starting the application process using a suspend option;

creating memory space in the application process;

injecting the redirect code into the memory space of the application process; and

set an instruction pointer to the redirect code.

27. The method, as set forth in claim 23, wherein injecting a redirect code further comprises:

starting the application process using a suspend option;

creating memory space in the application process;

injecting the redirect code into the memory space of the application process; and

use a create remote thread function to execute the redirect code.

28. The method, as set forth in claim 23, wherein executing the redirect code comprises:

loading the redirect library of redirect functions;

determining a location of an import table replacement function in the redirect library; and

executing the import table replacement function.

29. The method, as set forth in claim 28, wherein loading the redirect library of redirect functions comprises loading a dynamic link library.

30. The method, as set forth in claim 28, wherein executing the import table replacement function comprises:

searching an import table of the application process for the set of target function calls; and

modifying the target function calls to reference redirect functions in the redirect library.

31. The method, as set forth in claim 28, wherein executing the import table replacement function comprises:

searching dynamic link libraries of the application process for the set of target function calls; and

modifying the target function calls to reference redirect functions in the redirect library.

32. The method, as set forth in claim 22, wherein intercepting at least one target function call comprises intercepting at least one socket function call.

33. The method, as set forth in claim 22, further comprising executing redirect functions to enable a secured access to a plurality of resources via a firewall.